AMENDMENTS TO THE SPECIFICATION

Please replace the title with the following amended title:

~~DETERMINATION OF ONE OR MORE VARIABLES TO RECEIVE~~ METHOD OF DETERMINING VALUE ~~CHANGES~~ CHANGE FOR PLACEMENT VARIABLE ~~IN LOCAL SEARCH SOLUTION OF INTEGER PROGRAMMING PROBLEM~~

Please replace the related applications section at page 1, lines 6-13, with the following amended section:

The following applications disclose related subject matter: U.S. Application No. ~~(Attorney Docket No. 200209179-1)~~10/627,324, filed ~~(on the same day as this application)~~ Jul. 25, 2003, and entitled, "Determining Placement of Distributed Application onto Distributed Resource Infrastructure"; and U.S. Application No. ~~(Attorney Docket No. 200209180-1)~~ 10/627,883, filed ~~(on the same day as this application)~~ Jul. 25, 2003, and entitled, "Incorporating Constraints and Preferences for Determining Placement of Distributed Application onto Distributed Resource Infrastructure"; the contents of all of which are hereby incorporated by reference.

Please replace the field of invention section at page 1, lines 16-19, with the following amended section:

The present invention relates to the field of ~~solving an integer programming problem. More particularly, the present invention relates to the field of solving an integer programming problem where a constraint may include a polynomial term of at least second order.~~ placing a distributed application onto a distributed resource infrastructure. More particularly, the present invention relates to the field of placing a distributed application onto a distributed resource infrastructure where the distributed application and the distributed resource infrastructure have arbitrary communication topologies.

Please replace the paragraph at page 1, lines 22-32, with the following amended paragraph:

An integer program can be used to model a resource allocation problem in which variables are assigned discrete values. ~~An integer~~ linear integer program expresses a particular resource allocation problem as a set of linear equations or inequalities. A method of solving ~~an integer~~ linear integer program employs a local search solution. The local search solution uses a gradient following approach to iteratively improve an initial assignment of values to the variables until a near ~~optimum~~ optimal solution is reached. Each iteration of the local search solution produces a new assignment of values for the variables. Generally, the new assignment of values differs from a previous assignment of values by one value for a particular variable. For the ~~integer~~ linear integer program, a selection of the variables that follows the gradient is accomplished by evaluating coefficients for the variables in an unsatisfied constraint.

Please delete the paragraph at page 2, lines 16-17, which begins: "What is needed..."

Please insert the following paragraphs at page 2, line 16:

A distributed application includes a plurality of services. Each of the services performs a task or tasks as part of the distributed application. Often the distributed application is placed on a network of computers. The network of computers forms a distributed resource infrastructure where each of the computers forms a node. Performance of the distributed application depends on optimizing a placement of the services onto the nodes.

An existing method uses parameters for individual nodes to determine a placement of the services onto the nodes. Such parameters include processing and storage capabilities of the nodes. Services are placed onto the nodes so that processing and storage requirements of the services on a particular node do not the processing and storage capabilities of the node.

The existing method does not consider relationships among the nodes or among the services in the determination of the placement of the services onto the nodes. Another existing method considers topologies between the services and between the nodes but requires that the topologies be fixed in certain configurations. The other existing method does not determine a placement of the services onto the nodes where an arbitrary topology exists between the nodes or between the services.

Please replace the summary of the invention at page 2, lines 22-31, with the following amended summary of the invention:

The present invention is a method of determining ~~a variable to receive~~ a value change for a placement variable as part of a local search solution to an integer programming problem that models a placement of services of a distributed application onto nodes of a distributed resource infrastructure. ~~The method can be used where a constraint has one or more polynomial terms of at least second order.~~ In an embodiment, ~~of the present invention~~ an unsatisfied communication constraint is selected. Stores are created for allowable changes of value for ~~the~~ placement variables in the unsatisfied communication constraint. The unsatisfied communication constraint is parsed through by term. For each variable in a term, the stores are updated with a change in the term for each of the allowable changes of the value while maintaining other variables constant. ~~A variable to receive the~~ The value change~~, and possibly a value~~ for the placement variable~~,~~ ~~are~~ is chosen based upon the store which meets at least one improvement criterion.

~~These and other aspects of the present invention are described in more detail herein.~~

Please replace the brief description of the drawings section at page 3, lines 2-13, with the following amended section:

The present invention is described with respect to particular exemplary embodiments thereof and reference is accordingly made to the drawings in which:

Figure 1 illustrates a preferred method of determining a variable to receive a value change as a block diagram according to an ~~aspect~~ embodiment of the present invention;

Figure 2 illustrates a first alternative method as a flow chart according to an ~~aspect~~ embodiment of the present invention, where the first alternative method employs a local search solution to solve an integer programming problem that includes a polynomial term of at least second order;

Figure 3 illustrates an alternative step of choosing a variable to receive a value change according to an ~~aspect~~ embodiment of the present invention; ~~and~~

Figure 4 illustrates a system for determining a variable to receive a value change according to an ~~aspect~~ embodiment of the present ~~invention.~~ invention;

Figure 5 schematically illustrates an exemplary distributed application according to an embodiment of the present invention;

Figure 6 schematically illustrates an exemplary distributed resource infrastructure according to an embodiment of the present invention;

Figure 7 illustrates a method of determining a placement of services of a distributed application onto a distributed resource infrastructure as a block diagram according to an embodiment of the present invention;

Figure 8 illustrates an alternative method of determining placement of services of a distributed application onto a distributed resource infrastructure as a block diagram according to an embodiment of the present invention;

Figure 9 schematically illustrates an exemplary distributed application according to an embodiment of the present invention; and

Figure 10 schematically illustrates an exemplary distributed resource infrastructure according to an embodiment of the present invention.

Please replace the paragraph at page 3, lines 16-24, with the following amended paragraph:

The present invention is a method of ~~choosing a variable to receive~~ ~~determining~~ a value change for a placement variable as part of a local search solution to an integer programming problem. A constraint may include one or more polynomial terms of at

least second order (i.e., a quadratic or higher order polynomial term). The local search solution uses a gradient following approach to iteratively improve an initial assignment of values to the variables until a satisfactory solution is reached. Each iteration of the local search solution produces a new assignment of values for the variables. Generally, the new assignment of values differs from a previous assignment of values by one value for a ~~particular~~ placement variable.

Please insert the following paragraphs at page 11, line 3:

An embodiment of an exemplary distributed application is illustrated schematically in Figure 5. The distributed application 500 comprises a firewall 502, a local buffer 504, first, second, and third web servers, 506, 508, and 510, an application server 512, and a database 514, each of which forms a service. The firewall 502 is coupled to the local buffer 504. Each of the first, second, and third web servers, 506, 508, and 510, couples the local buffer 504 to the application server 512. The application server 512 is coupled to the database 514. In operation, the distributed application embodiment 500 provides web access to users who provide and obtain information from the database 514.

An embodiment of an exemplary distributed resource infrastructure is illustrated schematically in Figure 6. The distributed resource infrastructure 600 comprises first through eighth nodes, 602..616, and a switch 618. The switch 618 couples each of the first through eighth nodes, 602..616, to remaining nodes of the first through eighth nodes, 602..616. The determination of the placement of the distributed application 500 onto the distributed resource infrastructure 600 is accomplished according to an objective such as minimizing network traffic, minimizing latency in order to reduce response time, or balancing a load on the nodes.

An embodiment of a method of determining placement of services of a distributed application onto nodes of a distributed resource infrastructure of the present invention is illustrated as a block diagram in Figure 7. The method 700 comprises first through third steps, 702..706. The first step 702 forms communication constraints, which ensure that transport demands between node pairs do not exceed transport capacities between the

node pairs. Each of the communication constraints is made up of a sum of terms. Each of the terms comprises a product of a first placement variable, a second placement variable, and the transport demand between the services associated with the first and second placement variables. The second step 704 forms the objective. In the third step 706, a local search solution is employed to solve an integer program comprising the communication constraints and the objective, which provides the placement of the service onto the nodes.

A first alternative method of determining placement of services of a distributed application onto nodes of a distributed resource infrastructure of the present invention is illustrated as a block diagram in Figure 8. The first alternative method 800 adds fourth and fifth steps to the preferred method 700. The fourth step 802 establishes an application model, which comprise the transport demands between the services. The fifth step 804 establishes an infrastructure model, which comprises the transport capacities between the nodes. The application model and the infrastructure model provide the transport demands and capacities to the preferred method 700.

Another embodiment of an exemplary distributed application is illustrated schematically in Figure 9. The distributed application 900 comprises first through fourth services, 901..904, and fifth through Sth services, 905. Mathematically, the first through Sth services, 901..905, are expressed as $s \in \{1, 2, 3, ..., S\}$. Each pair of the services has an associated transport demand. For example, a first transport demand dt_{12} represents communication traffic between the first and second services, 901 and 902. A transport demand matrix Dt lists the transport demands between the first through Sth services, 901..905, as follows.

$$Dt = \begin{pmatrix} - & dt_{12} & dt_{13} & .. & dt_{1S} \\ dt_{21} & - & dt_{23} & .. & dt_{2S} \\ dt_{31} & dt_{32} & - & .. & dt_{3S} \\ .. & .. & .. & - & .. \\ dt_{S1} & dt_{S3} & dt_{S3} & .. & - \end{pmatrix}$$

Since services do not communicate with themselves over a network, the transport demands along a matrix diagonal have no values. Further, depending upon a particular implementation it may be sufficient to characterize the transport demands without

reference to direction in which case the transport demands below the matrix diagonal would also have no values.

Each of the first through Sth services, 901..905, of the alternative distributed application 900 is also characterized with a processing demand and a storage demand. For example, the first service 901 has a first processing demand dp_1 and a first storage demand ds_1. A processing demand vector Dp and a storage demand vector D_s list the processing demands and the storage demands of the first through Sth servers, 901..905, as follows.

$$Dp = \begin{pmatrix} dp_1 \\ dp_2 \\ dp_3 \\ .. \\ dp_S \end{pmatrix} \qquad Ds = \begin{pmatrix} ds_1 \\ ds_2 \\ ds_3 \\ .. \\ ds_S \end{pmatrix}$$

Another embodiment of an exemplary distributed resource infrastructure is illustrated schematically in Figure 10. The distributed resource infrastructure 1000 comprises first through fourth nodes, 1001..1004, and fifth through Nth nodes, 1005. Mathematically, the fist through Nth nodes are expressed as $n \in \{1, 2, 3, ..., N\}$. Each pair of the nodes has an associated transport capacity. For example, a first transport capacity ct_{12} represents communication bandwidth between the first and second nodes, 1001 and 1002. A transport capacity matrix Ct lists the transport capacities between the first through Nth nodes, 1001..1005, as follows.

$$Ct = \begin{pmatrix} - & ct_{12} & ct_{13} & .. & ct_{1N} \\ ct_{21} & - & ct_{23} & .. & ct_{2N} \\ ct_{31} & ct_{32} & - & .. & dt_{3N} \\ .. & .. & .. & - & .. \\ ct_{N1} & ct_{N3} & ct_{N3} & .. & - \end{pmatrix}$$

Since nodes do not communicate with themselves over a network, the transport capacities along a matrix diagonal have no values. Further, depending upon a particular implementation it may be sufficient to characterize the transport capacities without reference to direction in which case the transport capacities below the matrix diagonal would also have no values.

Each of the first through Nth nodes, 1001..1005, of the alternative distributed resource infrastructure 1000 is also characterized with a processing capacity and a storage capacity. For example, the first node 1001 has a first processing capacity cp_1 and a first storage capacity cs_1. A processing capacity vector Cp and a storage capacity vector Cs list the processing capacities and the storage capacities of the first through Nth nodes, 1001..1005, as follows.

$$Cp = \begin{pmatrix} cp_1 \\ cp_2 \\ cp_3 \\ .. \\ cp_N \end{pmatrix} \qquad Cs = \begin{pmatrix} cs_1 \\ cs_2 \\ cs_3 \\ .. \\ cs_N \end{pmatrix}$$

In some situations, the distributed application under consideration operates solely on the distributed resource infrastructure. In this situation, the transport, processing, and storage capacities represent absolute capacities for the nodes. In other situations, the distributed application is one of a plurality of distributed applications operating on the distributed resource infrastructure. In these other situations, the transport, processing, and storage capacities represent available capacities for the nodes.

In the alternative distributed application embodiment 900 and the alternative distributed resource infrastructure embodiment 1000, the transport and storage demands, Dt and Ds, as well as the transport and storage capacity, Ct and Cs, are normalized according to standard parameters of data per unit time and data, respectively. In an embodiment of the present invention, the processing demand Dp and the processing capacity Cp are normalized according to a processing criterion. Preferably, the processing criterion is a transaction speed especially when the distributed application forms a database application. Alternatively, the processing criterion is an algorithm speed. In another embodiment of the present invention, various processors are listed in a look-up table with associated processing capacities that have been normalized by the processing criterion. In this embodiment, a system implementing the present invention would go to the look-up table to find the processing capacity for a particular node when needed.

Applying the method 800 (Figure 8) to the distributed application 900 and the distributed resource infrastructure 1000 begins with the fourth and fifth steps, 802 and 804, in which the transport demand and capacity matrices, Dt and Ct, are established. The first step 702 then forms the communication constraints in first and second tasks. The first task forms placement variables, which according to an embodiment of the present invention are Boolean variables. In a matrix notation, the placement variables are given by a placement variable matrix X where rows represent the services and columns represent the nodes. The placement variable matrix X is as follows.

$$X = \begin{pmatrix} x_{11} & x_{12} & .. & x_{1N} \\ x_{21} & x_{22} & .. & x_{2N} \\ .. & .. & .. & .. \\ x_{S1} & x_{S2} & .. & x_{SN} \end{pmatrix}$$

The second task forms the communication constraints according to a communication constraint equation, which is given as follows.

$$\sum_{k} \sum_{i} x_{ij}\, x_{kl}\, dt_{ik} = ct_{jl}$$

The second step 704 forms the objective, which according to an embodiment of the present invention minimizes communication traffic between the nodes and balances processing loads on the nodes. The latter is accomplished by minimizing the mathematical variance of the processing loads. The objective is given as follows.

$$\text{Minimize } (1-\alpha)\frac{1}{A}\sum_{l}\sum_{j} Prox_{lj} \sum_{k}\sum_{i} x_{ij}\, x_{kl}\, dt_{ik} +$$

$$\alpha\{\sum_{j} (\frac{1}{cp_i}\sum_{i} x_{ij}\, dp_i)^2 - \frac{1}{N}\sum_{l} (\frac{1}{cp_l}\sum_{k} x_{kl}\, dp_k)^2 \}$$

for where α provides a relative weight between minimizing the communication traffic and balancing the processing loads, A provides a normalizing factor, $Prox_{lj}$ accounts for distances between the nodes, and N is a number of the nodes. The third step 706 then employs the local search solution to solve the integer program comprising the communication constraints and the objective.

Since the communication constraints account for a distributed application topology according to the transport demands and for a distributed resource infrastructure

topology according to the transport capacities, the present invention allows arbitrary topologies for both the distributed application and the distributed resource infrastructure. This allows the present invention to be used for determining placement of applications onto infrastructures in wide variety of situations. Examples of such situations include placing an application onto nodes in a data center and placing a different application onto nodes in geographically distributed data centers.

Another embodiment of a method of determining placement of services of a distributed application onto nodes of a distributed resource infrastructure of the present invention adds processing constraints to the integer program. The processing constraints ensure that a sum of the processing demands for a specific node does not exceed the processing capacity of the specific node. In an embodiment of the present invention, the processing constraints are formed according to a processing constraint equation, which is given as follows.

$$\sum_i x_{ij}\, dp_i \leq cp_j$$

Another embodiment of a method of determining placement of services of a distributed application onto nodes of a distributed resource infrastructure of the present invention adds storage constraints to the integer program. The storage constraints ensure that a sum of the storage demands for a specific node does not exceed the storage capacity of the specific node. In an embodiment of the present invention, the storage constraints are formed according to a storage constraint equation, which is given as follows.

$$\sum_i x_{ij}\, ds_i \leq cs_j$$

Another embodiment of a method of determining placement of services of a distributed application onto nodes of a distributed resource infrastructure of the present invention adds placement constraints to the integer program. The placement constraints ensure that each of the services is placed on one and only one of the nodes. In an embodiment of the present invention, the placement constraints are formed according to a placement constraint equation, which is given as follows.

$$\sum_i x_{ij} = 1$$

Another embodiment of a method of determining placement of services of a distributed application onto nodes of a distributed resource infrastructure of the present invention recognizes that, once the services have been placed onto the nodes, a rearrangement of the services onto the nodes comes at a cost. In the fifth alternative method, reassignment penalties are assessed when a service placement differs from an existing assignment of the service. According to an embodiment of the fifth alternative method, a second objective is added to the integer program. The second objective seeks to minimize the reassignment penalties.